Skills Embeddings: a Neural Approach to Multicomponent Representations of Students and Tasks

Russell Moore*, Andrew Caines*, Mark Elliott*†, Ahmed Zaidi*, Andrew Rice* and Paula Buttery*
ALTA Institute
*Computer Laboratory †Cambridge Assessment
University of Cambridge
{rjm49 | apc38 | mwe24 | ahz22 | acr31 | pjb48}@cam.ac.uk

ABSTRACT

Educational systems use models of student skill to inform decision-making processes. Defining such models manually is challenging due to the large number of relevant factors. We propose learning multidimensional representations (embeddings) from student activity data – these are fixed-length real vectors with three desirable characteristics: co-location of similar students and items in a vector space; magnitude increases with skill, and that absence of a skill can be represented. Based on the Multicomponent Latent Trait Model, we use a neural network with complementary trainable weights to learn these embeddings by backpropagation. We evaluate using synthetic student activity data that provides a ground-truth of student skills in order to understand the impact of number of students, question items and knowledge components in the domain. We find that our data-mined parameter values can recreate the synthetic datasets up to the accuracy of the model that generated them, for domains with up to 10 simultaneously active knowledge components, which can be effectively mined using relatively small quantities of data (1000 students, 100 items). We describe a procedure to estimate the number of components in a domain, and propose a component-masking logic mechanism that improves performance on high-dimensional datasets.

Keywords

knowledge representation, skills embeddings, multicomponent latent trait model

1. INTRODUCTION

Intelligent tutoring systems (ITS) are required to make decisions about which tasks to present to which students. Thus they should be equipped with objective, accurate models of student skillsets, to inform these choices. Student activity logs are a source of information for such models, which could be built by hand-crafted feature extraction, or using data mining methods. We explore the latter, using machine-learning of fixed-width multidimensional representations. We call these SKILLS EMBEDDINGS, after *word embeddings* – vector representations of words and language constructs that have allowed dramatic advances in the natural language processing field [6, 8].

With word embeddings, semantically similar words are positioned near each other in a latent vector space: *e.g,* 'boat' should be nearer to 'car' than 'politics', based on natural language data. We transfer the vector space idea to skills. For the ITS scenario, we seek specific desirable traits:

- Skills embeddings are **co-proximal** in the vector space if they represent entities comprising similar skills.
- Embedding **magnitude** grows with skill – specifically, a higher skill level should entail a larger value within the embedding. This lets us track skill gain intuitively.
- It should be possible to represent the special case where a skill is absent. We will refer to this characteristic as **skill masking**.

We would also ideally like to be able to **detect dimensionality**: the number and dependency structure of skills within the domain should not need to be specified in advance.

In this work, we propose an artificial neural network – based on the Multicomponent Latent Trait Model [12] – to learn skills embeddings. We train it using synthetic student activity datasets, with a varying number of skills in the domain. We show that the embeddings exhibit our three desired characteristics, and present a procedure to cater to the fourth.

2. BACKGROUND

Our work relies on some core principles about the nature of knowledge domains, the way that student ability and item difficulty interact, and the idea that knowledge acquisition can be traced in student activity logs [5].

2.1 Knowledge components

For any given educational domain, such as physics, mathematics or language learning, we can break domain-specific knowledge down into atomistic units known as KNOWLEDGE COMPONENTS (KCs), as described by Koedinger *et al.* [4]. We treat these as synonymous with 'skill' where the skill is irreducible – if skill C comprises irreducible subskills A and B, we do not represent C, but treat co-occurrence of A and B as the pattern for C. We think of a subject domain as a

Russell Moore, Andrew Caines, Mark Elliott, Ahmed Zaidi, Andrew Rice and Paula Buttery "Skills Embeddings: a Neural Approach to Multicomponent Representations of Students and Tasks" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*, Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 360 - 365

set K of KCs to be acquired. In our datasets, domain size $|K| \in [1, 100]$.

2.2 Rasch model

The Rasch item response model [11] is a well-known formulation for the success probability of student s attempting item (question) i, derived by transforming the difference between student ability θ_s and item difficulty β_i through a sigmoid function. That is, the probability of success is given by:

$$Pr(X_{si} = 1 \mid \theta_s, \beta_i) = \sigma(\theta_s, \beta_i) \tag{1}$$

where σ is the standard logistic sigmoid function:

$$\sigma(\theta_s, \beta_i) = \frac{1}{1 + \exp\left(-(\theta_s - \beta_i)\right)} \tag{2}$$

Note that an evenly-matched student-item pair has $\theta = \beta$ and a pass-rate of 0.5. The Rasch model assumes a single dimension of proficiency and embodies *invariant comparison* – this means the student parameter θ can be eliminated algebraically during estimation of the item parameters β, and vice versa [9, 14]. This principle allows Rasch items (and by extension our embeddings) to stand alone as objective representations, independent of the conditions in which they were measured.

2.3 Multicomponent Latent Trait Model

The Rasch model can be extended to the MULTICOMPONENT LATENT TRAIT MODEL (MLTM) of Whitely [12]. Here, the scalars θ and β are replaced by vectors, and the result is a product of sigmoids. The formulation is as follows:

$$Pr(X_{si} = 1 \mid \boldsymbol{\theta_s}, \boldsymbol{\beta_i}) = \prod_{k \in skills(i)} \sigma(\theta_{sk}, \beta_{ik}) \tag{3}$$

Hence the act of student s successfully passing item i is modelled as the conjunction of successes at each of the item's problem-solving steps (denoted k). The probability is given by the product of the probabilities of completing the steps and each step behaves as a Rasch model whose parameters are the corresponding elements of $\boldsymbol{\theta_s}$ and $\boldsymbol{\beta_i}$.

2.4 Item calibration

Traditionally, item calibration with Rasch-type models is carried out using the Birnbaum iteration [2]. However, the Birnbaum algorithm is one-dimensional and to the best of our knowledge has not been extended to multiple dimensions. Moreover, the Rasch approach does not readily allow for the absence of skills: parameters would have to be set to -∞, which is impractical for data mining, particularly in cases where the skill is absent both from a question and student's representations[1]

2.5 Q-Matrix

The Q-MATRIX [10, 1] is a binary- or probability-valued matrix that describes which skills are required for particular tasks. This has been used previously, for instance, in the Linear Logic Test Model [13]. Each column of $\mathbf{Q}$ represents a task/item i, and each row a component $k \in K$.

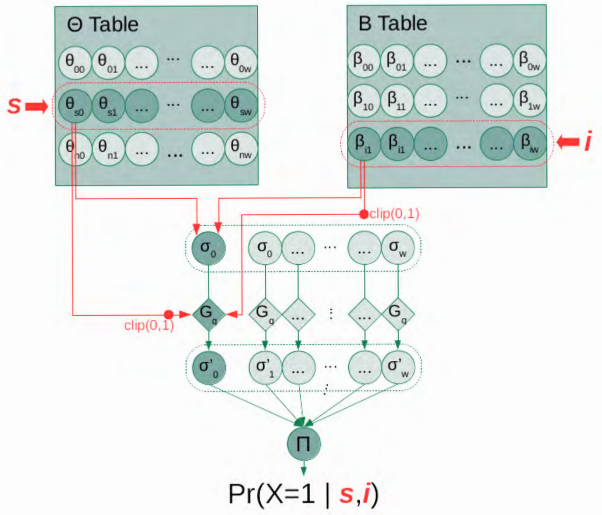

Figure 1: Neural network architecture

The (binary) Q-matrix for a curriculum of items is as follows:

$$q_{ik} = \begin{cases} 1 & \text{if } k \in skills(i), \\ 0 & \text{else;} \end{cases} \tag{4}$$

The Q-matrix for students is similar, with element q_{sk} representing the ability of student s at skill k.

3. DATA

Our datasets follow a *summative* assessment scenario: the item bank is a static test, to be attempted by many students. Each student attempts all items only once. Each response is dichotomous: either right (X=1) or wrong (X=0).

Student activity data is synthesised using a statistical model whose probability mass function (pmf) is just equation (3). The response x_{si} (of student s on item i) is determined by ground-truth values of the MLTM parameters, $\boldsymbol{\theta_s^*}$ and $\boldsymbol{\beta_i^*}$. These are the targets we hope to recover from the dichotomous outcome data: we want our embeddings to converge on these values.

The elements of $\boldsymbol{\theta_s^*}$ and $\boldsymbol{\beta_i^*}$ themselves are generated uniformly randomly for each student and item – their minimum and maximum values are chosen by an earlier randomised search process, that looks for suitable bounds to generate a balanced dataset given a specific dimensionality $|K|$.

We generated datasets with dimensionalities $|K| \in \{1, 2, 5, 10, 100\}$. We also created datasets where only a subset of components are active – for these datasets, the active components are chosen at random. We use |I|=100 items and |S|=1000 students.

4. IMPLEMENTATION

In this section we describe the neural network, including its design features, software used, and training approach.

[1] Even assuming infinite arithmetic is allowable, if θ and β are both -∞ then $(\theta - \beta) = 0$ and the probability of success is calculated as 0.5; in fact, for an unrequired skill, it should always be 1, since an unrequired step is always 'passed'.

4.1 Neural network architecture

The neural network in this work is a binary classifier. In a normal supervised learning task, a classifier takes an input set of features $\Phi(x)$ and class label C and learns the probability that a datapoint x is in a given class: $Pr(x \in C \mid \Phi(x))$.

In the embedding generation task there is no feature mapping Φ. Instead we have datapoints of form $(s, i, pass \in \{T, F\})$, describing whether student s passed item i. Inputs to the neural network are s and i, and class label $pass$.

The 'features' themselves are learned internally, with two distinct sets of trainable weights. One set of weights ($\Theta \in \mathbb{R}^{|S| \times |K|}$) represents the students, the other ($B \in \mathbb{R}^{|I| \times |K|}$) the items. Whenever s occurs in a datapoint, the weights for s (synonymous with $\boldsymbol{\theta_s}$) are selected from the table, and the same happens for $\boldsymbol{\beta_i}$ when i occurs.

The weights are fed into a locally connected layer that represents the components of the MLTM. Each unit in the layer applies a sigmoid function to generate a per-component probability. The component probabilities are then multiplied to get the overall output probability. This is scored against the true $pass$ value using a loss function, and the error is backpropagated to the weights tables. The weights are re-used whenever s or i appear in a datapoint, so they are forced towards values which best fit all observations. The trained rows of weights serve as our fixed-width embeddings.

The architecture is illustrated in Figure 1. For clarity, the connections are shown only for a single component, but all components function in parallel in the same way. The diamonds on the diagram represent Q-gates, trainable logic components which we will now describe.

4.2 Q-gates – logic for absent components

In high-dimensional domains, items usually do not exercise all skill components simultaneously. For instance, in both assessment and instruction, questions are usually designed to focus on a subset of skills. Rather than model the subset of skills explicitly, we iterate across the whole domain K and let a logic layer selectively deactivate components:

$$Pr(X_{si} = 1 \mid \boldsymbol{\theta_s}, \boldsymbol{\beta_i}) = \prod_{k \in K} G_q(q_{sk}, q_{ik}, \sigma(\theta_{sk}, \beta_{ik})) \qquad (5)$$

G_q is a *Q-gate*, a ternary logic gate related to logical implication, with the following truth table for each component per student, q_{sk}, and per item, q_{ik}:

q_{ik}	q_{sk}	G_q
1	1	$\sigma(\theta_{sk}, \beta_{ik})$
1	0	0
0	1	1
0	0	1

Q-gates are implemented as part of the neural network, and modify the component-level sigmoid outputs:

$$G_q(q_{sk}, q_{ik}, \sigma(\theta_{sk}, \beta_{ik})) = \sigma(\theta_{sk}, \beta_{ik}) q_{ik} q_{sk} + (1 - q_{ik}) \quad (6)$$

The correct values for q_{ik} and q_{sk} are learned during training. These *q-values* can either be stored in their own set of weights, or represented by special values in $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$. We use the latter technique with *weight clipping*, explained next.

4.3 Weight clipping

We employ *weight clipping* for two reasons: to ensure component positivity (so that vector magnitude must grow with skill), and to implement Q-gates. To ensure components take only positive values, weights are clipped to $[1, W]$, with large W to allow for changes in value during training. The range $[0, 1)$ is reserved to switch the component's Q-gate.

4.4 Training

The network was trained with a categorical cross-entropy loss function and the Adam optimiser [3]. Generally, training is fast, and a learning rate $\alpha \in [0.01, 0.1]$ is stable. Weight initialisation is significant for convergence speed and fit: a uniform random initialisation over $[\theta_{min}, \theta_{max}]$ for students and $[\beta_{min}, \beta_{max}]$ for items was found to work well. From all instances in the training set, 10% were randomly chosen for validation and to trigger early-stopping on $loss_{val}$ with $patience = 10$ (*i.e.* we wait for a better value for ten more epochs before quitting, keeping our best weights). This work was implemented in `Python 3.6` using `Keras` with a `TensorFlow` back-end, and `scikit-learn`.

5. EVALUATION

In this section we describe how the embeddings were evaluated *vis-à-vis* our desired characteristics.

5.1 Prediction agreement

We attempt to recreate the original datasets by using our embeddings $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\beta}}$ to seed our statistical MLTM model. We then score correlation and agreement between the outputs of the original process (seeded with targets $\boldsymbol{\theta^*}$ and $\boldsymbol{\beta^*}$) and the embedding-seeded process.

Since our dataset is synthetic, we can directly access the probabilities that determine the outcomes, and thus can measure the Pearson's correlation between these and the predicted probabilities from the embeddings. This is:

$$\rho_{X,Y} = \frac{cov(P^*, \hat{P})}{sd(P^*), sd(\hat{P})} \qquad (7)$$

where P^* and $\hat{P}$ are the true and predicted probabilities of a pass, $cov(\cdot, \cdot)$ is covariance and $sd(\cdot)$ is standard deviation.

Because the generator process is stochastic, there will always be some element of chance in the observed outcomes. *Cohen's Kappa* gives a measure of the agreement beyond chance. For N datapoints, where n_{agreed} is the observed agreement between both runs, and $n_{(k;seed)}$ is the number classed as category k by the model seeded with $seed$:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \text{ where } \begin{cases} p_o = n_{agreed}/N \\ p_e = \frac{1}{N^2} \sum\limits_{k=\{T,F\}} n_{(k;\boldsymbol{\theta^*}\boldsymbol{\beta^*})} n_{(k;\hat{\boldsymbol{\theta}}\hat{\boldsymbol{\beta}})} \end{cases} \qquad (8)$$

5.2 Co-proximity & Magnitude

To assess our co-proximity requirement, we measure Euclidean distance from the aligned embedding $\hat{\boldsymbol{\theta}}$ (or $\hat{\boldsymbol{\beta}}$) to its target, $\boldsymbol{\theta^*}$ (or $\boldsymbol{\beta^*}$). We compare this to the mean distance from other vectors in the space, and test for significance to show that co-proximity to target is not merely by chance.

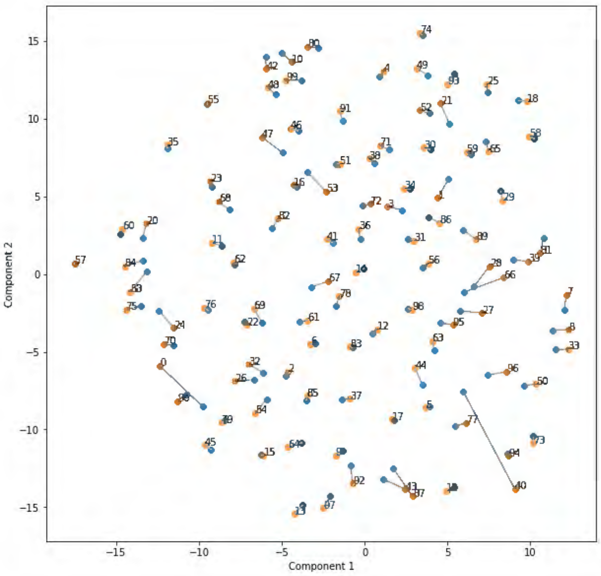

Figure 2: t-SNE visualisation of embeddings in a 10-dimensional space, all active, with 1000 students and 100 items (orange=embedding, blue=target).

To assess magnitude growth with skill, we use Pearson correlation (equation 7) at the component level between the elements of $\hat{\boldsymbol{\theta}}$ (or $\hat{\boldsymbol{\beta}}$) and of target $\boldsymbol{\theta}^*$ (or $\boldsymbol{\beta}^*$). A strong correlation shows that these values grow together as desired.

5.3 Aligning the components

Embeddings are identifiable only up to the ordering of the components due to conjunctive commutivity: columns in the Θ-Table will be aligned with the B-Table, since they were trained together, but they may be permutated differently to the columns in the original target vectors. To visualise the data, or calculate deviations from the true parameters, we must first align the predicted components. A hill-climbing algorithm can find the order needed to minimise the per-column squared error between the predicted and true values. While not guaranteed to find the global optimum, it is nonetheless reliable. Once the components are aligned, the embeddings can be plotted (after dimensionality reduction such as PCA or t-SNE for $|K| > 2$). The mean absolute parameter errors are given as θ_{MAE} and β_{MAE} in Table 1(b).

Figure 3 shows the mined values for a 2 KC domain, with 1-2 active components: Q-gates allow components to be switched on/off. Dotted lines show the thresholds below which the Q-gate treats a component as inactive. Figure 4 shows the mined values for a 10 KC domain, with 1-3 active components. Here the clustering of target points (blue) are more pronounced than in the all-active data (Figure 2). The embeddings (orange) cluster close to their targets due to the Q-gate mechanism.

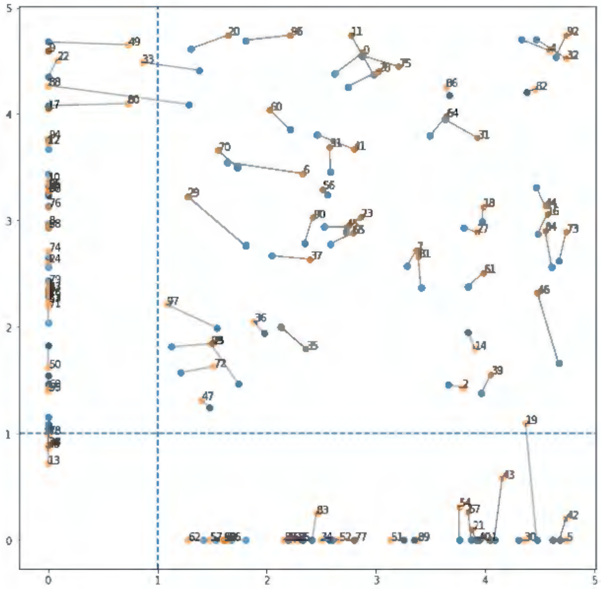

Figure 3: Direct plot of Q-gated embeddings in a 2-dimensional space, 1000 students and 100 mixed items (50×2-components, 50×1-component). Dotted lines show the Q-gate activation regions.

6. RESULTS & DISCUSSION

Table 1 gives summary statistics for all-active (upper section) and Q-gated (lower section) datasets.

Tabke 1(a) gives raw accuracy and Cohen's κ measure agreement between true and predicted outcomes, and Pearson's ρ measures correlation between the underlying probabilities. The reproduction (*repro*) scores for *Acc* and κ show how well the original model can reproduce its own data – this indicates the level of stochastic noise in the target dataset.

Reproduction accuracy drops from 0.844 to 0.651 as domain size increases, but this is more pronounced (0.684 to 0.310) for κ, implying that much of the accuracy score is down to chance, and κ is a more useful measure. At low to moderate dimensions (1a to 10a), mined κ values approach the model's own agreement, but for the high dimension (100a) they do not even achieve half of this limit, although still better than chance ($\kappa = 0$). For Q-gated data (2q2, 10q3, 10q5, 100q5) the embeddings produce higher accuracy and κ throughout than their non-Q-gated counterparts.

Pearson's correlation on the underlying probabilities (unaffected by stochastic noise) is very good (>0.9) for all lower $|K|$ data (with or without Q-gates), but for (100a) this drops to 0.442 - the Q-gated version (100q5) scores 0.754, an improvement of over 70%.

6.1 Co-proximity & Magnitude

Table 1(c) gives mean Euclidean distances of embedding to target, alongside the mean distance to other points in the dataset. Mean and standard deviations are given along with Welch's t-test results. In all cases the mined vectors have significantly (p<0.01) smaller mean distances to target than to other vectors in the space, indicating co-proximity between

Table 1: (a) Accuracy, Cohen's κ agreement, Pearson's correlation (ρ). The *repro* scores show how well the original dataset generator agrees with itself between runs, giving an upper limit to the score. (b) Parameter level error scores for students and items. (c) Mean Euclidean distance to target (D_{target}) and to other vectors (D_{others}), with t-test scores.

Model		(a) Model fit			(b) Param. fit.		(c) Co-proximity in vector space		
Name	Dims	Acc (repro)	κ (repro)	ρ^*	θ_{MAE}	β_{MAE}	D_{target}	D_{others}	t^{**}
1a	1	0.844 (0.844)	0.680 (0.684)	0.994	0.29	0.11	0.11 (0.09)	3.66 (0.65)	-36.9
2a	2	0.776 (0.777)	0.550 (0.556)	0.986	0.55	0.46	0.79 (0.38)	3.74 (0.80)	-33.1
5a	5	0.732 (0.742)	0.429 (0.439)	0.961	0.94	0.79	2.51 (1.36)	8.63 (1.24)	-33.0
10a	10	0.721 (0.733)	0.425 (0.463)	0.930	4.61	3.42	13.56 (3.32)	16.47 (1.06)	-8.26
100a	100	0.572 (0.651)	0.151 (0.310)	0.442	7.25	6.36	79.31 (5.40)	88.30 (2.18)	-15.0
2q2	2 (1-2)	0.801 (0.806)	0.576 (0.587)	0.987	0.58	0.41	0.24 (0.17)	4.76 (0.71)	-61.2
10q3	10 (1-3)	0.771 (0.802)	0.561 (0.588)	0.943	1.30	0.13	0.85 (0.63)	9.18 (1.49)	-51.2
10q5	10 (1-5)	0.822 (0.832)	0.503 (0.522)	0.942	1.33	0.19	1.12 (0.84)	10.53 (1.48)	-54.9
100q5	100 (1-5)	0.732 (0.821)	0.509 (0.555)	0.754	3.09	0.27	1.13 (0.84)	10.53 (1.48)	-54.0

*(p<0.01) **(Welch's t-test, df=98, p<0.01)

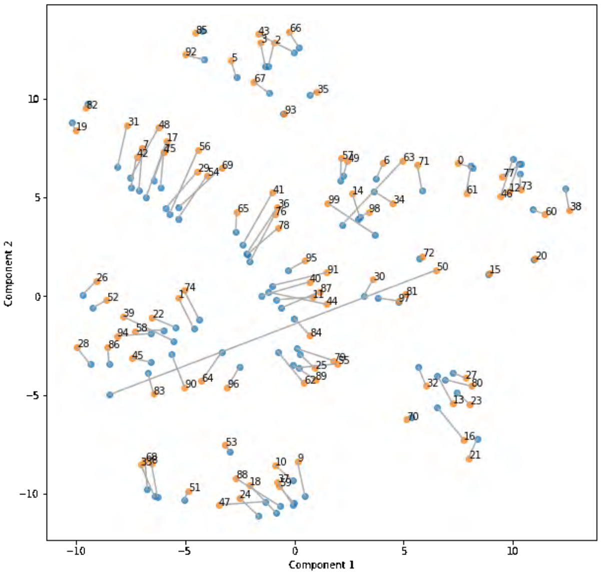

Figure 4: t-SNE visualisation showing 100 items, calibrated from 1000 students. Here, only 1-3 dimensions (from 10) are active for any item. The other dimensions are masked using Q-gates.

mined and true vectors as desired. Again Q-gates have a notable effect: for instance, the quotient (D_{others}/D_{target}) increases from 1.11 for 100a to 9.30 for 100q5.

Component-level correlations for true and mined parameter values (not tabled) are strongly correlated ($\rho \geq 0.90, p = 0.01$) for low to moderate $|K| \in [1, 10]$ showing our second desired characteristic of magnitude growth with skill. As with other results, a weaker positive correlation ($\rho = 0.21, p = 0.01$) was measured for $|K| = 100$.

6.2 Skill masking

Our embeddings achieve a high correlation with target outcome probabilities for all but $|K| = 100$. Similar patterns can be seen with other scores. Overall, larger error in the vector space (100a in Table 1(b)) seems to contribute to markedly reduced model fit (100a in Table 1(a)).

There are at least two factors at play here: firstly, for large $|K|$, the 'curse of dimensionality' makes distance metrics less meaningful, and it becomes difficult to determine distance (or similarity) between vectors. The second factor is informational: if an item has 100 active components, a student must achieve a pass-rate of 99.3% *on each component* to get 50% pass-rate on the item. Very easy components carry little information: our expectation that a student would pass them is almost always met. This manifests as a shallow gradient on the sigmoid in this region ($x = 4.95$), making parameter values very sensitive to stochastic noise in training data. Furthermore, dichotomous results tell us nothing about which component caused a failed attempt. These factors make for a difficult machine-learning task.

Fortunately, the Q-gated datasets show a different pattern. They behave more like low-dimensionality data: for instance, 100 (1-5 active) dimensions – *versus* 100 (all active) – shows both better probability correlation ($0.75 > 0.44$), and component level error ($3.09 < 7.52$ for $\boldsymbol{\theta}$ and $0.27 < 6.36$ for $\boldsymbol{\beta}$). A similar effect is seen in the 10-dimensional data.

The ability to mask off certain components is useful. For instance, $|K| = 100$ may be a reasonable domain size, but items will often have far fewer active skills, *e.g.* Pardos *et al.* [7] used a question-set for high-school mathematics with $|K| = 105$, but a maximum of three skills per question. Masking is vital to represent such a domain as fixed-width vectors. Moreover, with Q-gates, the exact number and composition of skills per question need not be known: it can be learned during training. Hence the width of our embeddings need not exactly match the domain: if they are too wide, the Q-gates will trim excess components. We give a procedure to estimate $|K|$ next.

6.3 Dimensionality estimation

Although it is not possible to directly detect the dimensionality $|K|$ of a domain, there is a simple procedure to estimate it. Embeddings are trained across a span of candidate values $|K|_{cand}$ and the mean maximum accuracy for each $|K|_{cand}$

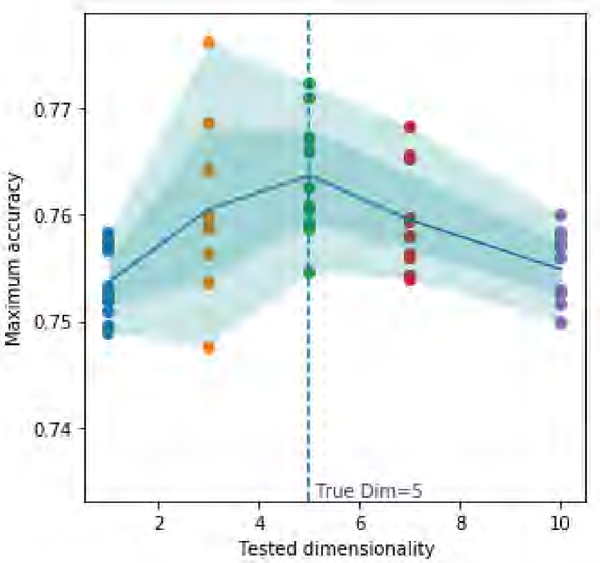

Figure 5: Dimensionality detection – candidate dimensions on x-axis, maximum accuracy for each fit on y-axis; ten runs per candidate, line plot shows mean maximum accuracy.

is calculated. Figure 5 shows the result of this process for a dataset with $|K| = 5$, $|S| = 1000$, $|I| = 100$. We plot candidates $|K|_{cand} \in \{1, 3, 5, 7, 10\}$ against the mean maximum accuracy (over 10 repetitions) of fit to the dataset. The peak at $|K|_{cand} = 5$ reveals the true value of $|K|$.

7. FUTURE WORK

We are in the process of mining skills embeddings from a major online physics-teaching platform. We believe the embeddings discovered will reveal more about realistic dimensionality and skill composition, and help us to study changes in student ability over time. We also intend to report on the interpretability of embeddings by human experts.

8. CONCLUSION

This work introduces a new technique to mine SKILLS EMBEDDINGS – student and item vector representations based on the Multicomponent Latent Trait Model – using a neural network with complementary weights. This was applied to synthesised student activity datasets, to recover the original seed parameters. We were able to extract these parameters in moderately high-dimension data ($|K|$=10) even for small datasets (100 items, 1000 students).

We gave four desired characteristics for our embeddings: co-proximity of similar objects in vector space, growth of magnitude with skill, ability to model missing skills, and applicability in domains of unknown dimension. We showed our embeddings support all but the fourth, and gave a procedure to mitigate this. We introduced Q-GATES, a skill masking mechanism that boosts model fit for high dimensional domains with realistic contraints.

9. ACKNOWLEDGMENTS

This paper reports on research supported by Cambridge Assessment, University of Cambridge. We thank the Isaac Physics team, our colleagues in the ALTA Institute, and the three anonymous reviewers for their valuable feedback.

10. REFERENCES

[1] M. Birenbaum, A. Kelly, and K. Tatsuoka. Diagnosing knowledge states in algebra using the rule-space model. *Journal for Research in Mathematics Education*, 24:442–459, 1993.

[2] A. Birnbaum. Some latent trait models and their use in inferring an examinee's ability. In F. M. Lord and M. R. Novick, editors, *Statistical Theories of Mental Test Scores*. Reading, MA: Addison-Wesley, 1968.

[3] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[4] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, 36(5):757–798, 2012.

[5] K. R. Koedinger, S. D'Mello, E. A. McLaughlin, Z. A. Pardos, and C. P. Rosé. Data mining and education. *Wiley Interdisciplinary Reviews: Cognitive Science*, 6(4):333–353, 2015.

[6] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[7] Z. Pardos, N. Heffernan, C. Ruiz, and J. Beck. The composition effect: Conjuntive or compensatory? an analysis of multi-skill math questions in its. In *Educational Data Mining 2008*, 2008.

[8] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[9] G. Rasch. *Probabilistic Models for Some Intelligence and Attainment Tests.* Danmarks Paedogogiske Institut, Copenhagen, 1960.

[10] K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, 20:345–354, 1983.

[11] W. J. van der Linden and R. K. Hambleton. *Handbook of Modern Item Response Theory.* Springer Science & Business Media, 2013.

[12] S. E. Whitely. Multicomponent latent trait models for ability tests. *Psychometrika*, 45(4):479–494, 1980.

[13] M. Wilson and P. De Boeck. Descriptive and explanatory item response models. In P. De Boeck and M. Wilson, editors, *Explanatory Item Response Models: A Generalized Linear and Nonlinear Approach.* New York: Springer-Verlag, 2004.

[14] B. D. Wright and J. M. Linacre. Dichotomous rasch model derived from specific objectivity. *Rasch Measurement Transactions*, 1:5–6, 1987.